REMARKS

No claims have been added, amended, or cancelled. Hence, Claims 25-49 are currently pending in the application.

SUMMARY OF INTERVIEW

The Examiner is thanked for the interview that occurred on April 14, 2005 between the Examiner, the corresponding Primary Examiner, and the Applicants' attorney, Chris J. Brokaw. During the interview, the following topics were discussed: (a) background of the Applicants' application, (b) the approach of *Ladd*, and (c) the features of Claim 1 which are not disclosed, taught, or suggested by *Ladd*. No agreement was reached. The Applicants submit herein the arguments that were presented during the interview.

SUMMARY OF THE REJECTIONS

Claims 25-49 are rejected under 35 U.S.C. § 102(a) as allegedly being anticipated by Ladd, Eric, et al., Using HTML 4, XML, and Java 1.2, 1999, Que. Platinum Edition (hereinafter "*Ladd*"). Claims 32 and 44 are rejected under 35 U.S.C. § 103(a) as allegedly being obvious in view of *Ladd*.

The rejections are respectfully traversed.

LADD DOES NOT SUGGEST EACH ELEMENT OF THE CLAIMS

Each of Claims 25-49 is patentable over the cited art because at least one element in each pending claim is not disclosed, taught, or suggested by the cited art.

Claim 25 recites:

"storing a preconstructed web page;
storing, separate from said preconstructed web page, correlation data that specifies a correlation between an identifier and replacement content;
receiving a request for a requested web page that corresponds to said preconstructed web page;
in response to said request, retrieving said preconstructed web page, wherein:
said preconstructed web page was created prior to receiving said request,
said preconstructed web page is written in a tag-delimited page description language, and
said preconstructed web page includes said identifier that is located at a position between a pair of tags within said preconstructed web page;
in response to said request, modifying said preconstructed web page to produce said requested web page by causing a program to perform the steps of:
removing said identifier from said preconstructed web page, and
inserting said replacement content at said position in said preconstructed web page, wherein said replacement content is selected based on the correlation data; and
providing said requested web page in response to said request." (emphasis added)

The above-cited combination of elements are not disclosed, taught, or suggested by *Ladd*.

Both *Ladd* and the pending claims are directed towards generating a web page, and both *Ladd* and the pending claims perform operations that modify a web page in response to receiving a request for a web page. However, beyond these broad generalities, there are sharp contrasts between the approach of *Ladd* and the approach recited in the pending claims.

The approach of *Ladd* is directed towards an Active Server Page (ASP) application that comprises scripted instructions embedded in an HTML document (see page 850 of *Ladd*). In the approach of *Ladd*, when the server receives a request for the web page, the server may execute a script embedded in the web page. The script may modify the content of the web page. For example, if a client requested the web page shown on page 851, then the server

would execute the embedded VBScript entitled 'HelloWorld,' which would generate the text "Hello World." The text "Hello World" would appear in the requested web page served to the client, but the embedded VBScript would not.

Unfortunately, the approach of *Ladd* suffers from exactly the same deficiencies that were described in the Applicants' background. Specifically, page 3, lines 5-16 of the Applicants' specification state:

> A second approach employs the reverse technique. Source code in a higher order programming or scripting language is placed into a Web page HTML code and is interpreted at runtime. Examples of this approach include ColdFusion, **ASP (Active Server Pages)** and JSP (Java Server Pages). Source code segments are placed directly into the HTML code. Tags or similar delimiters separate the source code segments from the page script. At runtime, a server executes the source code to generate a complete Web page. However, the source code is intrusive and can make using conventional Web editing tools difficult. The look and feel of a Web site can also be spread across several pages, imposing a maintenance burden. Finally, the application logic and user interface are not cleanly separated. Consequently, programmers and Web page artists find simultaneously working on the same page difficult. (emphasis added)

The approach of *Ladd* is directed towards the very same technique identified in the Applicants' background quoted above, namely Active Server Pages. Specifically, both the approach of *Ladd* and the portion of the Applicants' background quoted above feature the following same features:

1. Source code in a higher order programming or scripting language is placed into a Web page HTML code and is interpreted at runtime.
2. The scripting language or source code is an Active Server Page application.
3. Source code segments are placed directly into the HTML code.
4. Tags or similar delimiters separate the source code segments from the page script.

5. At runtime, a server executes the source code to generate a complete Web page.

6. The source code is intrusive and can make using conventional Web editing tools difficult.

7. The look and feel of a Web site can also be spread across several pages, imposing a maintenance burden.

8. The application logic and user interface are not cleanly separated.

Advantageously, the approach taken by the pending claims solves the problems described by the Applicants' background and experienced by *Ladd*. In the approach of the pending claims, correlation data that specifies a correlation between an identifier and replacement content is stored separate from the preconstructed web page. The replacement content, identified by the correlation data, is inserted into the preconstructed web page at a position identified by an identifier. Because the correlation data is stored separate from the preconstructed web page, the problems associated with the approach of *Ladd* are avoided.

To illustrate, FIG. 3 of the Applicants' patent application illustrates a pre-constructed web page. At lines 11 and 12 of FIG. 3, the identifiers #HFEF#, #SYMBOL#, and #COMPANY# are recited. When the pre-constructed page is requested, a controller script 31 (shown in FIG. 2) removes the identifiers, and inserts replacement content in lieu of the corresponding identifier based on correlation data.

Correlation data is data that is stored, separate from the pre-constructed web page, which specifies a correlation between an identifier and replacement content. For example, controller script 31 may contain the correlation data, as shown below and on page 12, lines 5, 19 of the Applicants' specification:

For example, a controller script 31 written in PL/SQL to generate a dynamic Web page from the HTT template 35 shown in FIGURE 3 is as follows:

```
BEGIN
htt.get ('companies.html');
htt.sub ('HREF', 'http://www.acme.com);
htt.sub ('SYMBOL, 'ACME');
htt.sub ('HREF', 'ACME');
htt.sub ('COMPANY, 'ACME Corporation');
htt.break;
htt.sub ('HREF', 'http://www.orcl.com);
htt.sub ('SYMBOL, 'ORCL);
htt.sub ('COMPANY, 'Oracle Corporation);
htt.showpage;
END;
```

As shown above, the correlation data in the controller script 31 associates the identifier HREF with replacement content (http://www.acme.com and http://www.orcl.com), associates the identifier SYMBOL with replacement content (ACME and ORCL), and associated the identifier COMPANY with replacement content (ACME Corporation and Oracle Corporation). As taught in the Applicants' specification at page 11, line 24 – page 12, line 4, the correlation data may specify that more than one piece of replacement content may have a correlation to a particular identifier, as shown above. The requested web page dynamically generated by the controller script 31 in this example is shown in FIG. 4.

In view of the differences between *Ladd* and the approach of the pending claims, numerous elements featured in Claim 25 are not disclosed, taught, or suggested by *Ladd*. No portion of *Ladd* teaches, discloses, or suggests the element of "storing, separate from said preconstructed web page, correlation data that specifies a correlation between an identifier and replacement content" as featured in Claim 25. On the contrary, *Ladd* expressly teaches away from this element as the approach of *Ladd* stores all code that modifies web page content in the actual web page itself. For example, the script "HelloWorld," on page 851 of

Ladd, is contained within a particular web page, and when the particular web page is requested, the script "HelloWorld" is executed.

Significantly, in the approach of *Ladd*, to the extent that anything is analogous to an identifier as claimed, the identifier must be the embedded script itself in its entirety, since the script is removed from the served web page in the approach of *Ladd*, and Claim 25 requires that "removing said identifier from said preconstructed web page." Further, to the extent that anything is analogous to replacement content as claimed, the replacement content must be the result of processing the script embedded within the web page. For example, when the Hello World script of Listing 33.3 on page 851 of *Ladd* is executed, the only data specifying that "Hello World" is to be printed upon execution of the script is contained within the script.

However, in the approach of *Ladd*, no data is stored anywhere that specifies a correlation between (a) the embedded script in its entirety, and (b) the result of processing the embedded script. While the embedded script may be processed to generate replacement content, processing a script to generate a result is not analogous to storing data that specifies a correlation between the embedded script in its entirety and the replacement content generated by processing the embedded script. Consequently, *Ladd* cannot possibly disclose, teach, or suggest the feature of "storing, separate from said preconstructed web page, correlation data that specifies a correlation between an identifier and replacement content" as recited in Claim 25.

Assuming, *arguendo*, that *Ladd* does teach storing data analogous to correlation data, to the extent that the approach of *Ladd* contains anything remotely analogous to "correlation data that specifies a correlation between an identifier and replacement content" as claimed in Claim 25, it must be the script, embedded within the web page, either in its entirety or a portion thereof. However, Claim 25 also requires that the correlation data be stored "separate

from said preconstructed web page." In sharp contrast, the embedded script of *Ladd* is not separate from the preconstructed web page, but actually resides **inside** of the preconstructed web page.

As a result, the limitation of "storing, separate from said preconstructed web page, correlation data that specifies a correlation between an identifier and replacement content" featured in Claim 25 cannot possibly be disclosed, taught, or suggested by *Ladd*.

The Office Action argues "the correlation data is present in the superstructure of data and code that exists to implement ASP (Active Server Pages) on the machine in *Ladd*." The Applicants respectfully submit such a broad interpretation of correlation data cannot be maintained because, as expressly featured in Claim 25, "correlation data that specifies a correlation between an identifier and replacement content." While the Applicants readily acknowledge that an external entity may process a preconstructed web page, such as the preconstructed web page shown on page 851 of *Ladd*, no entity external to the preconstructed webpage specifies a correlation between an identifier and replacement content in the approach of *Ladd*.

Specifically, the "superstructure of data and code" that processes the embedded ASP code in the approach of *Ladd* does not specify any correlation between an identifier and replacement content. To be clear, the Office Action is arguing that the "superstructure of data and code" maintains a correlation between (a) the embedded script itself in its entirety (which is allegedly analogous to an identifier as claimed), and (b) the result of processing the VB script (which is allegedly analogous to replacement content as claimed). In other words, the "superstructure of code and data" must store data that describes a correlation between the embedded script and the result of processing the embedded script. However, if such a "superstructure of data and code" did operate as the Office Action alleges, then there would

be absolutely no reason to embed the script in the preconstructed web page in the first place, since the "superstructure of data and code" already stores data that identifies the result of processing the embedded script.

Further, to support the Office Action's argument, such a "superstructure of data and code" would need to be updated with the eventual result of processing each script that the "superstructure of data and code" is configured to execute. Such a requirement runs contrary to both (a) all known environments for executing scripts, and (b) the teachings of *Ladd*, which indicates: "ASPs are not precompiled. Rather, they are interpreted when they are requested by a browser" (See page 850 of *Ladd*). Thus, it is respectfully submitted that the Office Action's interpretation that "superstructure of data and code that exists to implement ASP" maintains correlation data as claimed cannot be supported.

Further, no portion of *Ladd* teaches, discloses, or suggests the element of "inserting said replacement content at said position in said preconstructed web page, wherein said replacement content is selected based on the correlation data" as featured in Claim 25. In sharp contrast, to the extent that content is inserted into a web page in the approach of *Ladd*, the content to be inserted is selected by data contained within the embedded script of the web page, and not by correlation data stored separate from the preconfigured web page.

Consequently, as at least one element is not shown, taught, or suggested by the cited art, it is respectfully submitted that Claim 25 is patentable over the cited art, and is in condition for allowance. Claims 26-49 contain elements similar to those discussed above with respect to Claim 25. Thus, Claims 26-49 are also in condition for allowance for at least the same reasons as given above for Claim 25.

CONCLUSION

The Applicants believe that all issues raised in the Office Action have been addressed and that allowance of the pending claims is appropriate. The Examiner is respectfully requested to contact the undersigned by telephone if it is believed that such contact would further the examination of the present application.

For the reasons set forth above, it is respectfully submitted that all of the pending claims are now in condition for allowance. Therefore, the issuance of a formal Notice of Allowance is believed next in order, and that action is most earnestly solicited.

To the extent necessary to make this reply timely filed, the Applicant petitions for an extension of time under 37 C.F.R. §1.136. If any applicable fee is missing or insufficient, throughout the pendency of this application, the Commissioner is hereby authorized to any applicable fees and to credit any overpayments to our Deposit Account No. 50-1302.

Respectfully submitted,
HICKMAN PALERMO TRUONG & BECKER LLP

Date: April 20, 2005

Christopher J. Brokaw
Reg. No. 45,620

2055 Gateway Place, Suite 225
San Jose, CA 95110
Telephone: (408) 414-1080, ext. 225
Facsimile: (408) 414-1076

CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Mail Stop Amendment, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

On 4/22/05 By Trudy Bagdon
Trudy Bagdon